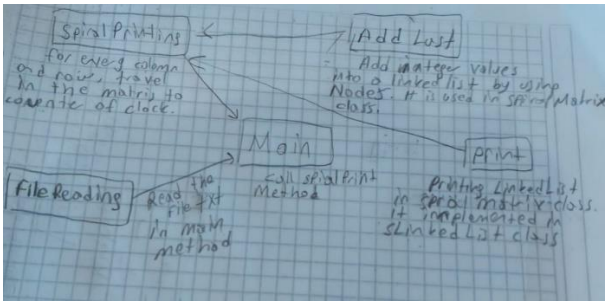
Giray Berk KUŞHAN SEC 04

ID: 10889878942

Q1A:

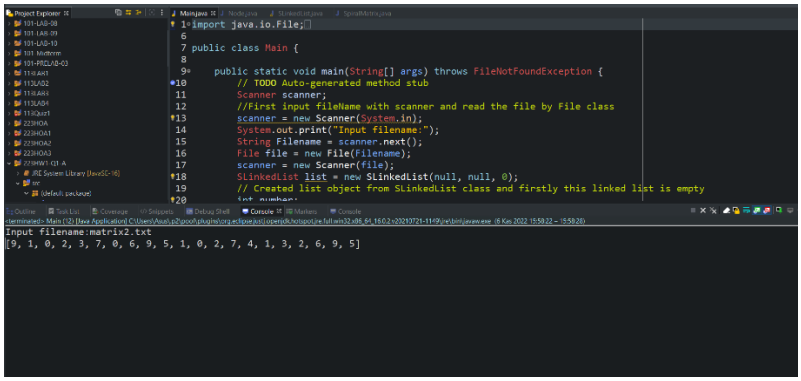
Problem Statement and Code Design:

In this assignment I created a project that travel spiral matrix which direction of counter of clock. The aim of this program is to read matrix files and travel in matrix array. But when we face with –1, Stop traveling. Then add them into a LinkedList and print that list. Code scheme of that design is in bottom line.



İmplementation and Functionality:

This project is created from 4 classes. These names are SpiralMatrix, SlLinkedList, Main and Node classes.



Output of matrix2.txt

Traveling in Spiral Matrix:

I used 2D array to travel in spiral matrix. But hard part is, travel direction is counter of clock. So, I used different integer variables to keep first row last row and columns. And I detect the size multiple of row and column. And I started to travel in this matrix. I did this step for every row and column. Then when the matrix is face to -1, I stopped in there.

Add LinkedList:

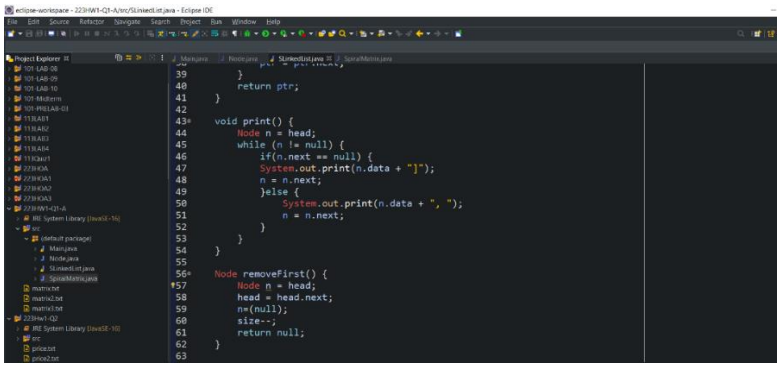
While I am travelling, I Used AddLast method for every index. In addLast method I controlled Is Empty method in if condition then I started to add using node.

Print List:

This is a last step, I printed matrix by using PrintList method in SLinkedList class. In this method I controlled list is null or not then I started to print each index in while loop.

Read File:

In main class, I have taken an Input as a String. Then I put into a file object which is created by File class. Then I detected row size in while loop and for every increasing number. I detected column size to num / row.



```
39          }
40          return ptr;
41      }
42
43      void print() {
44          Node n = head;
45          while (n != null) {
46              if(n.next == null) {
47              System.out.print(n.data + "]");
48              n = n.next;
49              }else {
50                  System.out.print(n.data + ", ");
51                  n = n.next;
52              }
53          }
54      }
55
56      Node removeFirst() {
57          Node n = head;
58          head = head.next;
59          n=(null);
60          size--;
61          return null;
62      }
```

```
62      }
63
64      void addLast(Node n) {
65
66          if (isEmpty()) {
67              n.next=(null);
68              head = tail = n;
69          } else {
70              tail.next=(n);
71              n.next=(null);
72              tail = n;
73              size++;
74          }
75      }
```

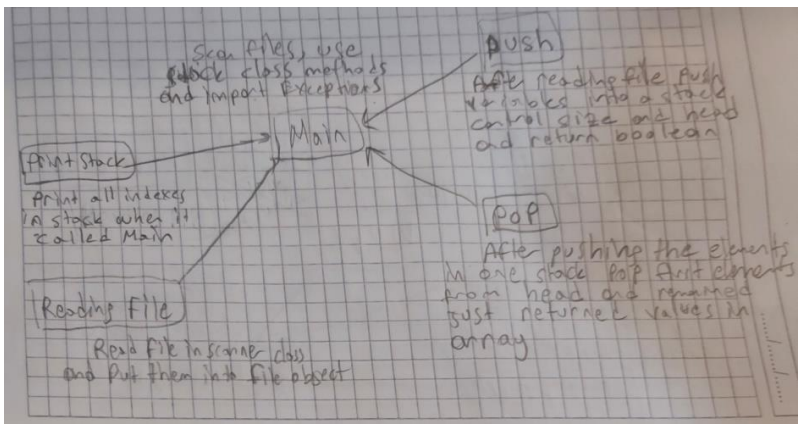
Testing:

There are three distinct test cases in programming homework. If 2D arrays have same size of row and column. It would be much easier for me. Because when I wrote my first algorithm in Q1A, it could not work properly int matrix2.txt since its size is bigger than test1. So, I had to wrote more general code. Because of test2. But test3 is easier than others because -1 is in the first index.

Q2:

Problem Statement and Code Design:

In this part of assignment, I preferred creating a stack class to control array because using stack has many advantages whereas queue. Using pop and push methods are easier than queue because we can push or pop in head or tail in Stack. Code scheme of that design is in bottom line.



In this project is occurred by two classes as main and Stack. First of all, I defined a integer array and I used pop and push methods in this array and print them.

Reading File:

In main class first I enter a String variable by using Scanner class, then I used into an File class. And in while loop I put them into an array.

Pushing:

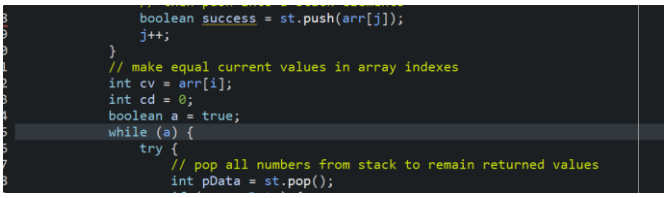
in stack class I controlled size and while I was pushing into stack, I increased the size.

Pop:

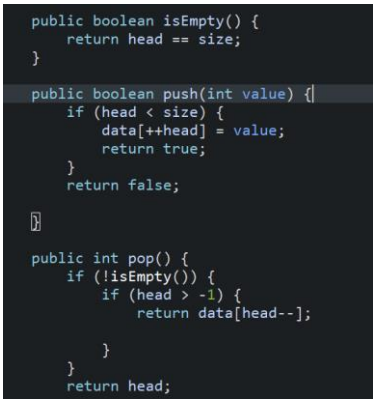
As I mentioned push, I checked stack is empty whether by using isEmpty method

Controlled And Printing:

Create while loop and check array length and used push method. Then I equalled every consecutive day in my array indexes and I assigned a integer to control that operation and for all pop increased cd and I printed stack by using printStack method.



```
                boolean success = st.push(arr[j]);
                j++;
            }
            // make equal current values in array indexes
            int cv = arr[i];
            int cd = 0;
            boolean a = true;
            while (a) {
                try {
                    // pop all numbers from stack to remain returned values
                    int pData = st.pop();
```



```
public boolean isEmpty() {
    return head == size;
}

public boolean push(int value) {
    if (head < size) {
        data[++head] = value;
        return true;
    }
    return false;

}

public int pop() {
    if (!isEmpty()) {
        if (head > -1) {
            return data[head--];

        }
    }
    return head;
```

Testing:

There are 3 distinct test cases, and they were not too hard for me I used general stack algorithm.

Final Assessments:

1. I had trouble when I was trying to true algorithm in this project. While I was researching true algorithm, I had learned many new information. And I have found proper solution.
2. I have learned Stack and LinkedList structure While I was trying to find true way.
3. The hardest example of me in all of questions, traveling in matrix2 because its size is bigger than matrix3 and matrix one. So, begging of the code I used to Boolean types and while I am travelling in indexes these types return to true. But because of matrix2 I had have to change this method.